ANSWERS

F8.10 Project: send a picture

BugBotLab

Data representation · GCSE · about 25 min

What this lesson is about

Capture a camera frame, make it 1-bit, compress it with RLE and send it by radio.

- The brief
- Step 1: capture and convert
- Step 2: compress it
- Step 3: decode and check
- Step 4: send it

Questions

4 marks in all

1. A 32 × 24 image at 1 bit per pixel. How many bits uncompressed? [1 mark]

> **Answer:** 768. 32 × 24 × 1.

2. Why are the RLE rows joined with `/`? [1 mark]

- ☐ **A** So the receiver knows where each row ends
- ☐ **B** To make the message shorter
- ☐ **C** Because RLE requires it
- ☐ **D** To encrypt the picture

> **Answer:** A. The separator is metadata the receiver needs to rebuild the rows.

3. How does the program prove the compression was lossless? [1 mark]

- ☐ **A** It decodes the message and checks the rows equal the original picture
- ☐ **B** It checks the message is under 200 characters
- ☐ **C** It counts the # characters
- ☐ **D** It sends the message twice

> **Answer:** A. Lossless means the decoded data is exactly the original.

4. Why does a busier picture make a longer message? [1 mark]

- ☐ **A** It has shorter runs, so RLE saves less
- ☐ **B** It has more pixels
- ☐ **C** Busy pictures use more colours in 1-bit
- ☐ **D** The radio slows down

> **Answer:** A. RLE depends on long runs of the same value.

The task: send a picture

Build the program from the brief. Print the 24 picture lines, then `uncompressed: 768 bits` (calculated), `compressed: <n> characters`, and `lossless: True`. Send the message if it has 200 characters or fewer.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

img = camera_image(32, 24)
```

The hint students can ask for: Turn the camera frame into rows of two characters, compress each row, and join the rows with a separator the receiver can split on. Check the message fits the radio limit, and prove it is lossless by decoding your own message and comparing.

A solution

```
from bugbot import *
connect()
def rle_encode(text):
    out = ""
    count = 1
    for i in range(1, len(text)):
        if text[i] == text[i - 1]:
            count = count + 1
        else:
            out = out + str(count) + text[i - 1]
            count = 1
    return out + str(count) + text[-1]

def rle_decode(code):
    out = ""
    number = ""
    for ch in code:
        if ch.isdigit():
            number = number + ch
        else:
            out = out + ch * int(number)
            number = ""
    return out
img = camera_image(32, 24)
picture = []
for row in img:
    line = ""
    for r, g, b in row:
        line = line + ("#" if (r + g + b) / 3 < 128 else ".")
    picture.append(line)
for line in picture:
    print(line)
print("uncompressed:", 32 * 24, "bits")
message = "/".join(rle_encode(line) for line in picture)
print("compressed:", len(message), "characters")
rows = [rle_decode(part) for part in message.split("/")]
print("lossless:", rows == picture)
if len(message) <= 200:
    send(message)
```

```
else:
    print("too big")
```

Any program that meets the task's checks is marked correct in the simulator; this is one way, not the only way.